**1.Merge Sort**

#include <stdio.h>

void simplemerge(int a[], int low, int mid, int high) {

int i = low, j = mid + 1, k = 0;

int c[10];

while (i <= mid && j <= high) {

if (a[i] <= a[j]) {

c[k++] = a[i++];

} else {

c[k++] = a[j++];

}

}

while (i <= mid) {

c[k++] = a[i++];

}

while (j <= high) {

c[k++] = a[j++];

}

for (i = low, k = 0; i <= high; i++, k++) {

a[i] = c[k];

}

}

void mergesort(int a[], int low, int high) {

int mid;

if (low < high) {

mid = (low + high) / 2;

mergesort(a, low, mid);

mergesort(a, mid + 1, high);

simplemerge(a, low, mid, high);

}

}

Output:

Enter number of elements: 4

Enter array elements: 2 6 3 7

Sorted array:

2 3 6 7

**2.Quick Sort**

#include<stdio.h>

int partition(int a[10],int low,int high)

{

int i,j,pivot;

i=low;

j=high+1;

pivot=a[low];

while(i<=j)

{

do

{

i=i+1;

}while(a[i]<pivot && i<=high);

do

{

j=j-1;

}while(a[j]>pivot && j>=low);

if(i<j)

{

int temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

a[low]=a[j];

a[j]=pivot;

return j;

}

void quicksort(int a[10],int low,int high)

{

int mid;

if(low<high)

{

mid=partition(a,low,high);

quicksort(a,low,mid-1);

quicksort(a,mid+1,high);

}}

int main() {

int n, a[10], i, low, high;

printf("\nEnter number of elements: ");

scanf("%d", &n);

printf("\nEnter array elements: ");

for (i = 0; i < n; i++) {

scanf("%d", &a[i]);

}

low = 0;

high = n - 1;

quicksort(a, low, high);

printf("\nSorted array:\n");

for (i = 0; i < n; i++) {

printf("%d\t", a[i]);

}

printf("\n");

return 0;

}

Output:

Enter number of elements: 5

Enter array elements: 2 6 3 7 8

Sorted array:

2 3 6 7 8